





## IF YOU HAVE PROBLEMS

LDOS has been created as a powerful, flexible, and user-oriented system. If you do run into problems, before you pick up the phone, do this:

- 1) Read the IN CASE OF DIFFICULTY section and do the checks indicated there.
- 2) READ THE MANUAL.  
Check syntax and spelling carefully.  
Review notes and technical information.  
Verify your understanding of the purpose of the command.  
Check if any updated version is available that deals with the problem.
- 3) RETRY THE OPERATION.  
Repeat the procedure again.  
Reset (BOOT) and repeat the procedure again.  
Retry the operation using your Master Diskette, if possible.  
Perform the same function in a different manner, if possible (let the Utility prompt for information rather than putting it in the command line or vice-versa, don't abbreviate the parameter, remove unnecessary system options, etc.)
- 4) THINK.  
Did it work last time? If so, - what has changed since then?  
Could it be a faulty diskette? Maybe another copy would work.  
Is everything turned on, plugged in, etc?  
Is a needed file not present on the disk, such as a system file, data file, etc?
- 5) WRITE IT DOWN!  
Make notes on the problem, the things you have tried, and the exact steps that led to the problem. The more detailed the notes, the better!
- 6) CALL.  
Call the Customer Service number during the proper hours.





LDOS 5.1.4 includes many small but important changes from 5.1.3, including the addition of the FED and QFB utilities. Many of the other changes are detailed on the reverse of this page.

To update your existing diskettes, we recommend the following procedure:

- 1) Make several mirror-image backups of your new master system diskette, then put it in a safe place. Never use your master diskette for any purpose other than the production of working copies.
- 2) Boot with one of your new working system diskettes. Place your old system diskette in drive one, and execute the following command:

**BACKUP \$:0 :1 (OLD)**

This will transfer new versions of the updated files to your old diskette.

Note: With the exception of the special case noted below, do not attempt to use any configuration file created under 5.1.3. Create a new configuration file under 5.1.4 if desired. If your working diskettes are any older than 5.1.3, it is recommended that you prepare new working system disks from your 5.1.4 master, and then move your existing programs and data files over to these new diskettes.

- 3) Model 1 owners may now re-boot with the updated diskette and transfer any updated files from the LDOSXTRA diskette if desired.
- 4) If desired, the new utilities (QFB and FED) may now be copied to any diskette with sufficient free space.

#### **Special instructions for Radio Shack Hard Disk Users:**

If you are not familiar with creating a new configuration under 5.1.4, and haven't changed your configuration since it was created by the INIT procedure, you may use the following procedure to update your hard disk system.

- 1) Boot a backup of your new 5.1.4 diskette. Execute the following command:  
**SYSTEM (SYSGEN)**
- 2) Now, place the 5.1.4 diskette in drive 1, and re-boot with your old 5.1.3 startup disk. Execute the following commands:

**SYSTEM (SYSTEM=4)  
BACKUP :5 :4  
COPY CONFIG/SYS.CCC:0 :5**

- 3) The 5.1.4 diskette is now your new startup disk. Place a write-protect tab on it and make several backups.

### LDOS 5.1.4 Changes and Technical documentation

The 5.1.4 release includes all published patches to the 5.1.3 release previously listed in the LDOS Quarterly/LSI Journal. In addition, the following changes have been made:

For the Model 3 version, LDOS will take advantage of the higher system clock speed available when running on a Model 4 in the Model 3 mode. Because of this, the real-time-clock will run at twice normal speed, and cassette I/O will not be possible. The clock speed can be returned to normal for these Model 3 operations with the SYSTEM (SLOW) command. SYSTEM (FAST) will return the clock speed to the Model 4 rate. KI/DVR has been adjusted for better keyboard debounce in the fast mode.

Also for the Model 3 version, there have been changes to the floppy disk drivers to improve operation on newer FDC designs.

On the MAX-80, the drive motor start-up delay has been corrected to give the proper one-half and one second delay times.

For all versions:

The method of allocating disk space has been changed to allocate disk space sequentially starting at the lower numbered cylinders. If the old method of "random" allocation is desired, it may be restored with the following patch:

PATCH SYS8/SYS.SYSTEM (D00,FE=D5 CD C1 44 D1 6C) for Model 1  
PATCH SYS8/SYS.SYSTEM (D00,FE=D5 CD 4E 44 D1 6C) for Mod 3 and MAX-80

The DATE and TIME prompts on boot have been altered to accept any character less than a "0" as a delimiter between digits. This will allow entry entirely from the keypad using the "." as a delimiter (e.g. 10.01.83).

LCOMM has been corrected to solve a buffering problem during long sessions.

PATCH has been corrected to allow X-type patches that span sectors to now be fully YANKed.

Several small changes have been made to FORMAT to better handle drives that require additional settling time.

The A parameter of the DIR command now defaults to "ON". If the old default of "OFF" is preferred, the following patch may be applied:

PATCH SYS6/SYS.SYSTEM (D05,D9=~~00 00~~)

Two new utilities have been added to the system: FED and QFB. Complete documentation is included with this update. For the Model 1, these utilities are located on the LDOSXTRA disk.

The bootstrap step rate will have no effect on any drive except drive 0 - you must use the SYSTEM library command as previously explained to adjust your other drive step rates.

DELAY refers to the amount of time between the drive motor startup and the first attempted access. It is valid for floppy drives only; hard disk drive motors run all the time. Some delay is necessary to allow a floppy disk drive motor to come up to its normal speed. The default will be 1 second for the Model I, and .5 seconds for the Model III, and may be changed with the SYSTEM (DRIVE=,DELAY=) Library command.

### Special Floppy disk drive types

The "standard" LDOS floppy disk types are:

40 track, double density, single side - Model III  
35 track, single density, single side - Model I

LDOS will also support any track count up to 96, double density, and double sided drives PROVIDED YOU HAVE THE CORRECT HARDWARE! For the Model III, all that is needed to add double sided support is a double headed drive and the proper drive cable. No special driver programs are required.

For the Model I, double density requires a double density board and use of the PDUBL or RDUBL driver program. These drivers also provide support for double headed drives. For single density, double sided users, the TWOSIDE program must be used. Also, any time double headed drives are used, an appropriate drive cable is needed. The standard drive cable will NOT work.

### STANDARD DISK FORMATS

This table reflects the format used on standard LDOS data diskettes.

| Size | Density | Tracks | Sides | Sectors/<br>Granule | Granules/<br>Cylinder | Directory<br>Sectors | Total<br>Files | User<br>Files | Free<br>Space |
|------|---------|--------|-------|---------------------|-----------------------|----------------------|----------------|---------------|---------------|
| 5"   | Single  | 35     | 1     | 5                   | 2                     | 8                    | 64             | 48            | 84K           |
| 5"   | Single  | 35     | 2     | 5                   | 4                     | 18                   | 144            | 128           | 169K          |
| 5"   | Double  | 35     | 1     | 6                   | 3                     | 16                   | 128            | 112           | 152K          |
| 5"   | Double  | 35     | 2     | 6                   | 6                     | 32                   | 256            | 240           | 305K          |
| 5"   | Single  | 40     | 1     | 5                   | 2                     | 8                    | 64             | 48            | 96K           |
| 5"   | Single  | 40     | 2     | 5                   | 4                     | 18                   | 144            | 128           | 194K          |
| 5"   | Double  | 40     | 1     | 6                   | 3                     | 16                   | 128            | 112           | 174K          |
| 5"   | Double  | 40     | 2     | 6                   | 6                     | 32                   | 256            | 240           | 350K          |
| 5"   | Single  | 80     | 1     | 5                   | 2                     | 8                    | 64             | 48            | 196K          |
| 5"   | Single  | 80     | 2     | 5                   | 4                     | 18                   | 144            | 128           | 394K          |
| 5"   | Double  | 80     | 1     | 6                   | 3                     | 16                   | 128            | 112           | 354K          |
| 5"   | Double  | 80     | 2     | 6                   | 6                     | 32                   | 256            | 240           | 710K          |
| 8"   | Single  | 77     | 1     | 8                   | 2                     | 14                   | 112            | 96            | 302K          |
| 8"   | Single  | 77     | 2     | 8                   | 4                     | 30                   | 240            | 224           | 606K          |
| 8"   | Double  | 77     | 1     | 10                  | 3                     | 28                   | 224            | 208           | 568K          |
| 8"   | Double  | 77     | 2     | 10                  | 6                     | 32                   | 256            | 240           | 1138K         |

## MEMORY USAGE AND CONFIGURATION

Certain features of LDOS are user selectable (i.e. the keyboard driver KI/DVR, the printer filter PR/FLT, Model I double density driver, etc). To implement these features, LDOS will load the necessary program into high memory. There is one term that is very important in the LDOS operating system - HIGH\$.

This term is pronounced "High dollar", and refers to a location that holds the address of the highest unused memory location. If LDOS is using no high memory, then HIGH\$ will contain X'7FFF', X'BFFF', or X'FFFF' for 16K, 32K, and 48K machines, respectively. To see the current HIGH\$ value, use the MEMORY Library command. When LDOS needs to use high memory, it does so in the following manner:

- 1) Find the highest unused memory address by looking at the value stored in the HIGH\$ location.
- 2) Install the necessary code in memory below the current HIGH\$ value.
- 3) Lower the HIGH\$ value to protect the added program code.

Any code that LDOS stores in high memory is written to be relocatable. This means that it can load anywhere in memory, and is not restricted to a specific area. Since LDOS always respects the HIGH\$ value, it will never attempt to overlay any programs loaded and protected by using the HIGH\$ value in this manner.

Unfortunately, other operating systems and/or applications programs do not always respect the HIGH\$ value. As a result, programs or BASIC USR routines that load in high memory are not always written in a relocatable manner. They have a fixed load address, and MUST be loaded there to execute properly. If LDOS has previously put program code in that memory area, it will be overwritten. This results in what is called a "memory conflict" - two pieces of program code that want to use the same memory area at the same time. When the LDOS code is something like the KI/DVR program, this usually results in an immediate system crash.

Fortunately, it is possible to get around this problem by using the MEMORY Library command. To resolve a memory conflict, you need only to know the load address and length of the unrellocatable code. We will consider two cases - when the code loads at the very top of memory, and when it loads at some other point.

When the conflicting code loads at the very top of memory, it is very easy to resolve the problem. Since you know the load address of the code, use the MEMORY Library command to change the HIGH\$ value to one byte below that address. For example, if a piece of code loads from address X'F900' and goes to the top of memory, you would issue a MEMORY (HIGH=X'F8FF') command. LDOS will now put any of its own high memory code below X'F900', protecting the module that will load there.

When the conflicting code does not load at the top of memory, you can use the same method just described to protect it. However, this will waste any memory between the end of the program and the top of memory. Let's consider the case where a module loads at X'F200' and extends to X'F3FF'. There is 3K of space between the end of the module and the top of memory. To avoid wasting this space, use the following procedure.

- 1) Load an LDOS module into high memory (i.e., SET KI/DVR, install a filter, etc).
- 2) Type in the command MEMORY with no parameters to see the current HIGH\$ value.
- 3) If the HIGH\$ value is above X'F3FF', repeat steps 1 and 2. If the value has gone below X'F3FF', you will need to start over, stopping before you load the module that caused the HIGH\$ value to go below X'F3FF'.

## Disk Organization

Due to the limited amount of space available on a thirty-five track, single-density diskette, the Model 1 version of LDOS is supplied on two disks. The first disk (labeled LDOS) contains the operating system itself, and the second disk (labeled LDOSXTRA) has the rest of the utilities and files that would not fit on the first disk. The tables located on the following page show which files are contained on each disk. Model 3 owners will only have one disk (labeled LDOS), that contains all the Model 3 LDOS files.

Note that the files from the LDOSXTRA disk may be transferred to any other disk as long as sufficient free space is available. If this is done on a single drive system, the COPY command with the (X) parameter must be used. See COPY for more information. Note that the correct password must be supplied from Page 1 - 17.

On the LDOSXTRA disk, LX80/DCT and RS232L/DVR are files for owners of the LOBO Systems LX-80 Expansion Interface. These files can be deleted by users who do not own this unit.

Model 1 LDOS users capable of running double-density may create a double-density LDOS system disk containing all the Model 1 LDOS files. This is discussed in detail in the "Special Hardware" section that follows.

## Moving existing files to LDOS

Files created under other TRS-80 operating systems may, in most cases, be moved to LDOS diskettes. Files located on TRSDOS 2.3B (Model 1) and TRSDOS 1.3 (Model 3) may be moved directly by using the COPY23B/BAS and CONV/CMD utilities respectively.

Model 1 owners may COPY files directly from TRSDOS 2.3. Model 3 owners must use REPAIR :d (ALIEN) first, and may then COPY. Both Model 1 and 3 owners may COPY from thirty-five track, single density NEWDOS and DOSPLUS disks if they are REPAIRED first. Read REPAIR in section 3 before attempting any transfers.

Moving files from Model 1 TRSDOS 2.7DD, 2.8 or other operating systems or formats not noted above is best accomplished using a disk transfer utility designed for the purpose. Although not specifically endorsed by LSI, one example of such a utility is Super Utility Plus. For pricing, availability and details, contact PowerSoft at 11500 Stemmons Freeway, Suite 125, Dallas TX, 75229 (214)-484-2976. Please note that such a utility is often not required, but may be the most convenient method. In general, if it is not desired to obtain such a utility, then the following procedure should be used:

- 1) Format a disk as thirty-five track, single-density, single-sided under the "alien" operating system.
- 2) Copy the file to this diskette.
- 3) Re-boot under LDOS, and process the diskette with REPAIR :d (ALIEN).

The files may now be copied to any other LDOS disk.

# Files on the Model 1 LDOS system disk:

| Filespec      | Attributes | Prot | /   | LRL | #Recs | / | Ext | File  | Space     | Mod | Date |
|---------------|------------|------|-----|-----|-------|---|-----|-------|-----------|-----|------|
| BACKUP/CMD IP | EXEC       | /    | 256 | 21  | /     | 1 | S=  | 6.2K  | 01-Oct-83 |     |      |
| BASIC/CMD IP  | EXEC       | /    | 256 | 2   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| BOOT/SYS SIP  | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K  |           |     |      |
| DIR/SYS SIP   | READ       | /    | 256 | 10  | /     | 1 | S=  | 2.5K  |           |     |      |
| FORMAT/CMD IP | EXEC       | /    | 256 | 20  | /     | 1 | S=  | 5.0K  | 01-Oct-83 |     |      |
| KI/DVR IP     | EXEC       | /    | 256 | 6   | /     | 1 | S=  | 2.5K  | 01-Oct-83 |     |      |
| LBASIC/CMD IP | EXEC       | /    | 256 | 20  | /     | 1 | S=  | 5.0K  | 01-Oct-83 |     |      |
| LBASIC/OV1 IP | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| LBASIC/OV2 IP | EXEC       | /    | 256 | 7   | /     | 1 | S=  | 2.5K  | 01-Oct-83 |     |      |
| LBASIC/OV3 IP | EXEC       | /    | 256 | 6   | /     | 1 | S=  | 2.5K  | 01-Oct-83 |     |      |
| PATCH/CMD P   | EXEC       | /    | 256 | 9   | /     | 1 | S=  | 2.5K  | 01-Oct-83 |     |      |
| PDUBL/CMD P   | EXEC       | /    | 256 | 4   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| RDUBL/CMD P   | EXEC       | /    | 256 | 4   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS0/SYS SIP  | NO         | /    | 256 | 17  | /     | 1 | S=  | 5.0K  | 01-Oct-83 |     |      |
| SYS1/SYS SIP  | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS10/SYS SIP | NO         | /    | 256 | 2   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS11/SYS SIP | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS12/SYS SIP | NO         | /    | 256 | 4   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS2/SYS SIP  | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS3/SYS SIP  | NO         | /    | 256 | 3   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS4/SYS SIP  | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS5/SYS SIP  | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS6/SYS SIP  | NO         | /    | 256 | 52  | /     | 1 | S=  | 13.8K | 01-Oct-83 |     |      |
| SYS7/SYS SIP  | NO         | /    | 256 | 39  | /     | 1 | S=  | 10.0K | 01-Oct-83 |     |      |
| SYS8/SYS SIP  | NO         | /    | 256 | 3   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |
| SYS9/SYS SIP  | NO         | /    | 256 | 5   | /     | 1 | S=  | 1.2K  | 01-Oct-83 |     |      |

# Files on the Model 1 LDOSXTRA Disk:

| Filespec      | Attributes | Prot | /   | LRL | #Recs | / | Ext | File | Space     | Mod | Date |
|---------------|------------|------|-----|-----|-------|---|-----|------|-----------|-----|------|
| BOOT/SYS SIP  | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K |           |     |      |
| CMDFILE/CMD P | EXEC       | /    | 256 | 12  | /     | 1 | S=  | 3.8K | 01-Oct-83 |     |      |
| CONV/CMD IP   | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| COPY23B/BAS   | ALL        | /    | 256 | 2   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| DIR/SYS SIP   | READ       | /    | 256 | 10  | /     | 1 | S=  | 2.5K |           |     |      |
| EQUATE1/EQU   | ALL        | /    | 256 | 14  | /     | 1 | S=  | 3.8K | 01-Oct-83 |     |      |
| FED/CMD P     | EXEC       | /    | 256 | 30  | /     | 1 | S=  | 7.5K | 01-Oct-83 |     |      |
| JL/DVR P      | EXEC       | /    | 256 | 2   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| KSM/FLT P     | EXEC       | /    | 256 | 3   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| LCOMM/CMD P   | EXEC       | /    | 256 | 11  | /     | 1 | S=  | 3.8K | 01-Oct-83 |     |      |
| LOG/CMD P     | EXEC       | /    | 256 | 1   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| LX80/DCT P    | READ       | /    | 256 | 2   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| MINIDOS/FLT P | EXEC       | /    | 256 | 4   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| MOD1/DCT P    | READ       | /    | 256 | 2   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| PR/FLT P      | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| QFB/CMD P     | EXEC       | /    | 256 | 15  | /     | 1 | S=  | 3.8K | 01-Oct-83 |     |      |
| REPAIR/CMD IP | EXEC       | /    | 256 | 3   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| RS232L/DVR P  | EXEC       | /    | 256 | 5   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| RS232R/DVR P  | EXEC       | /    | 256 | 4   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |
| TWOSIDE/CMD P | EXEC       | /    | 256 | 2   | /     | 1 | S=  | 1.2K | 01-Oct-83 |     |      |

The following section has been produced to answer many of the questions that users have been asking about the LDOS system. Please review this section before calling LDOS Customer Support.

## Special Hardware

LDOS provides for a wide degree of compatability with non-Radio Shack hardware, as long as it conforms to or is upward-compatable with the design specifications of Radio Shack hardware. Here are some examples:

### Two-sided disk drives

LDOS supports double-sided disk drives on the TRS-80 Models 1 and 3. On both models, the hardware must be fully capable of operating in the double-sided mode. These capabilities include:

1) Double-sided drives. Drives should be configured for operation as a single physical volume, with side-select on pin 32.

2) A controller board capable of using pin 32 as side-select (Most RS equipment is capable of this). Note that on the Model 1, use of pin 32 as side-select precludes the possibility of using a fourth disk drive if any drive in the system is double-sided. Also on the Model 1, if pins 32 and 34 are joined together by a trace at the edge connector in the Expansion Interface, this trace should be cut to ensure proper operation.

3) A disk drive cable capable of double-sided operation. For this, the lead connecting pin 32 must be continuous throughout the cable. Radio Shack disk drive cables \*are not\*! The easiest way to achieve this is to obtain a disk drive cable with \*no\* pins removed or disconnected, and then program each drive for its proper drive select address.

If you have any questions regarding the above, please contact your disk drive vendor for more information.

A bootable double-sided system disk may be created for the Model 3 by using the following procedure:

1) Format the destination disk as double-sided. The FORMAT command should look something like this: `FORMAT :1 (SIDES=2,CYL=40)`

2) Now, issue the command `BACKUP SYS0:0 :1 (SYS)`  
The drive numbers used may be modified if necessary.

3) Last, execute the command `BACKUP :0 :1 (INV, SYS)`

The resulting diskette should boot, and can be backed-up (mirror-image) without having to follow this special procedure.

On the Model 1, a disk driver capable of operating double-sided drives must be loaded before attempting to use both sides of double-sided drives. Both RDUBL and PDUBL (discussed later) have the ability to operate double-sided drives. If no double-density adapter is being used, TWOSIDE may be loaded to allow

double-sided operation. Simply enter the TWOSIDE command at LDOS Ready, and then do a SYSTEM (SYSGEN). The function of TWOSIDE will now be present whenever that disk is booted. It will be necessary to copy the file TWOSIDE/CMD over from the LDOSXTRA disk first, if not already present on the system disk.

### Double-density on the Model 1

LDOS supports both the PERCOM-type and Radio Shack double-density adapters. The proper driver software must be loaded before the double-density mode can be used. For the RS-type adapter the driver is called RDUBL, and for the PERCOM-type it is called PDUBL. Most non-RS adapters will use the PDUBL driver, even if not manufactured by PERCOM. Simply enter either the RDUBL or PDUBL command at LDOS Ready. This will load the driver into memory. Again, the SYSTEM (SYSGEN) command may be used to automatically re-load this configuration when booting.

Please note that the Model 1 hardware does not allow a true double-density disk to be booted. Since LDOS uses a standard double-density format across all models and LDOS implementations, "split density" disks are not supported (TRSDOS 2.7DD is an example of such a disk). For ease of double-density use, it is best to use the SYSTEM (SYSGEN) command to store a double-density configuration on a bootable single-density system disk, and AUTO the LOG utility (LOG/CMD must be copied from the LDOSXTRA disk first). Here is the step-by-step procedure to achieve this configuration:

- 1) Make a backup of your system disk and place the backup in drive 0
- 2) Type either RDUBL or PDUBL, and press <enter>
- 3) Type SYSTEM (SYSGEN) and press <enter>
- 4) Insert the LDOSXTRA disk in drive 1
- 5) Type BACKUP LOG/CMD:1 :0 and press <enter>
- 6) Type AUTO LOG and press <enter>
- 7) Place a write-protect tab on the disk in drive 0. This is your boot disk.
- 8) Place a blank disk in drive 1 and format it for double-density. A typical command for this would be FORMAT :1 (CYL=40,DDEN)  
For more information, see FORMAT in section 3.
- 9) Type BACKUP :0 :1 and press <enter>
- 10) When the BACKUP is done, press reset. When prompted for "new system disk", remove the disk from drive 0 and insert the disk from drive 1. Then, place the LDOSXTRA disk in drive 1.
- 11) Type BACKUP :1 :0 and press <enter>. When the backup is done, the disk in drive 0 is a complete double-density LDOS system disk. Place a write-protect tab on it and back it up.

The result of this setup is that when the single-density disk is booted, you will be prompted to insert the "new system disk". At this point, remove the boot disk, and insert the double-density system disk. Press <enter> and the normal "LDOS Ready" prompt will appear.

### Eight inch disk drives and hard disks

LDOS supports these drives with the proper hardware modifications and driver software. Since the driver software is specific to the particular hardware involved, contact your hardware vendor for more information. An article concerning eight inch drive operation was published in the Volume 2 Number 3 issue of the LSI Journal.

### System Clock Speed

Limited support for clock speed-up kits is built into LDOS. The LDOS system is designed to be insensitive to clock speed as much as possible, and should function without change at speeds up to 5 or so MHz. The following system commands are available for affecting the system clock speed:

SYSTEM (FAST) will result in the system "fast clock" flag being set. On the Model 1, a X'01' will also be sent to port X'FE'. On the Model 3, the proper actions will be taken to utilize the higher clock speed available on the Model 4 when running in the Model 3 mode. Note that this is the default boot up configuration. The system clock speed may be returned to normal for cassette I/O and other speed-dependent operations with the below command.

SYSTEM (SLOW) will reset the system "fast clock" flag. On the Model 1, a X'00' will be sent to port X'FE'. On the Model 3, the clock speed of a Model 4 running in the Model 3 mode will be returned to normal Model 3 speed.

If it is necessary to alter the values or ports noted above, the following issues of the LSI Journal should be obtained: Volume 2 Number 1 and Volume 2 Number 4.

### Hardware Clock/Calendars

General patches to Model 1 LDOS for hardware clocks using the MSM5832 type of clock chip were presented in the Volume 2 Number 6 LSI Journal. Please note that this is a general patch, and modifications may be needed for different brands or types of clock devices.

No such patch is available for the Model 3, due to Model 3 ROM restrictions.



## FED - THE LDOS FILE EDITOR

FED is an all-purpose, screen oriented file editor to be used with the LDOS operating system. Its wide range of capabilities make it excellent for the advanced user, but its simplicity makes it easy to use for the novice. The editor supports both Model I and III, upper and lower case, and all drive types and sizes supported by LDOS. Some points need to be made concerning FED:

This is a file editor, NOT a file copier, text editor, or word processor. It is for displaying, printing, and modifying existing files. Fed works on a file level, not a track/sector level.

FED was not designed to repair damaged disks or recover lost files, but it could be used to do so by the experienced LDOS user.

You cannot create or extend files with FED, only modify existing ones.

FED is intended to run with the LDOS operating system only.

The following is a brief description of FED's capabilities:

- 1) Complete editing capabilities are supported, including Hexadecimal and ASCII modifying. Direct disk patching becomes a simple matter with FED. It is even possible to write machine language code directly to disk. Small changes in files can be made instantly. With FED, there is no need to read in a large source file and reassemble it just to change one character.
- 2) FED allows for record advancing, backspacing and positioning. You may page through a file quickly, either forward or backward. The user need not know any diskette information (density, number of sides, number of sectors per gran, etc.). The only thing that is required to use FED is knowledge of the proper filespec.
- 3) ASCII and Hex string searching can be performed, and a command exists which will allow you to position the cursor to the next occurrence of the search string. FED searches the entire file, not just the current edit record. It allows searching for upper/lower case ASCII strings (up to 30 characters in length), and Hex strings (up to 15 bytes in length). FED will retain a search string, so you can go to the next occurrence of that string from the currently displayed position in the file.
- 4) You will be allowed to locate a Hex load address in a load module format file, and calculate the load position of a specified byte. This feature will facilitate the inspection and editing of a load module file. Just type in the load address in question, and FED will position the display to that byte. Another extremely powerful feature is the reverse of the address location command. FED will calculate where in memory a specific byte pointed to by the cursor will load. With these two features it is possible to write machine language routines directly to disk. Direct patches are made quickly and easily. Even X-patches can be installed by the experienced programmer.

- 5) Complete listing of a file or individual record to a printer is supported. Many safeguards have been added to make it difficult to LOCK-UP the system if a printer is deselected, out of paper, etc.
- 6) FED includes a 256 byte display mode, and an extended 128 byte display. Editing utilities in the past allowed for 256 byte displays only. By using this format exclusively, the variations of an ASCII/HEX display are limited. But by having a 128 character display mode, the extra space makes it more visually appealing. The filespec, drivespec, record number, input & output can be displayed horizontally instead of vertically.

Here is a sample display of the 256 byte mode:

| ASCII<br>representation | Hexadecimal representation |       |      |      |      |      |      |      |      |     |                      |  |  |  |  |  | Current Record |  |  |  |  |  |  |  |  |
|-------------------------|----------------------------|-------|------|------|------|------|------|------|------|-----|----------------------|--|--|--|--|--|----------------|--|--|--|--|--|--|--|--|
| !.h..=.UX.S @...!       | 00>                        | 21D8  | 6811 | 003D | CD55 | 58ED | 5320 | 4006 | 1721 | 0F  | Filespec             |  |  |  |  |  | F              |  |  |  |  |  |  |  |  |
| .`.@...Zx. .>..3        | 10>                        | ED60  | CD40 | 00DA | 945A | 78B7 | 2005 | 3E13 | C333 | 0E  |                      |  |  |  |  |  | E              |  |  |  |  |  |  |  |  |
| `.X..`.D .!.b.          | 20>                        | 60CD  | 2058 | 11ED | 60CD | 1C44 | 20F0 | 210D | 6206 | 0D  |                      |  |  |  |  |  | D              |  |  |  |  |  |  |  |  |
| ..\$D.3`:.`.0*.Rw       | 30>                        | 00CD  | 2444 | C233 | 603A | F360 | C630 | 2A15 | 5277 | D/  | Extention            |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| .K.`..C.R....C.R        | 40>                        | ED4B  | F960 | 0BED | 4313 | 5201 | 0000 | ED43 | 0F52 | C   |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| !.a".R>..3..UU.:        | 50>                        | 210D  | 6122 | 0A52 | 3E1C | CD33 | 00CD | 5555 | C93A | M   |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| .R...Z....Z..`.B        | 60>                        | 0152  | B7CC | 105A | FE04 | D410 | 5A11 | ED60 | CD42 | D   | - Drive #            |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| D.3`..`.6D.3`!.b        | 70>                        | 44C2  | 3360 | 11ED | 60CD | 3644 | C233 | 6021 | 0D62 | :   |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| ..[.....[...(U.         | 80>                        | 110D  | 6101 | 0001 | EDB0 | CDE3 | 5BC9 | CD28 | 55CD | 5   |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| ..]..K.R...`.BD.        | 90>                        | DE5D  | C9ED | 4B0F | 52C5 | 11ED | 60D5 | CD42 | 44C2 |     | - Relative<br>- Byte |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| 3`.6D.3`!.b..a..        | A0>                        | 3360  | CD36 | 44C2 | 3360 | 210D | 6211 | 0D61 | 0100 |     |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| .....*.R..B...BD        | B0>                        | 01ED  | B0D1 | C12A | 1352 | B7ED | 42C8 | 03CD | 4244 |     |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| .3`.6...UD.3`.G:        | C0>                        | C233  | 60CD | 3601 | 028E | 5544 | C233 | 60C9 | 473A |     | Command              |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| .R.(.!.?" @.!.=6        | D0>                        | 0E52  | B728 | 0721 | CA3F | 2220 | 40C9 | 21BD | 3D36 |     |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| ..#6..@..6...6..+6      | E0>                        | 8C23  | 36AC | 1140 | 0019 | 36AA | 10FB | 3683 | 2B36 | >82 |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
| !.!=" @...R...!=        | F0>                        | 8321  | FD3D | 2220 | 40C9 | 3A0E | 52B7 | C021 | BD3D | C:R |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
|                         |                            | Index |      |      |      |      |      |      |      |     |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |
|                         |                            |       |      |      |      |      |      |      |      |     |                      |  |  |  |  |  |                |  |  |  |  |  |  |  |  |

Here is a sample display of the 128 byte mode:

.K.`..C.R....C.R!.a".R>..3..UU.:.R...Z....Z..`.BD.3`..`.6D.3`!.b - ASCII  
 ..[.....[...(U..]..K.R...`.BD.3`.6D.3`!.b..a.....\*.R..B...BD - Rep.

|     | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |        |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--------|
| 40> | ED | 4B | F9 | 60 | 0B | ED | 43 | 13 | 52 | 01 | 00 | 00 | ED | 43 | 0F | 52 | - Hex  |
| 50> | 21 | 0D | 61 | 22 | 0A | 52 | 3E | 1C | CD | 33 | 00 | CD | 55 | 55 | C9 | 3A | - Rep. |
| 60> | 01 | 52 | B7 | CC | 10 | 5A | FE | 04 | D4 | 10 | 5A | 11 | ED | 60 | CD | 42 |        |
| 70> | 44 | C2 | 33 | 60 | 11 | ED | 60 | CD | 36 | 44 | C2 | 33 | 60 | 21 | 0D | 62 |        |
| 80> | 11 | 0D | 61 | 01 | 00 | 01 | ED | B0 | CD | E3 | 5B | C9 | CD | 28 | 55 | CD |        |
| 90> | DE | 5D | C9 | ED | 4B | 0F | 52 | C5 | 11 | ED | 60 | D5 | CD | 42 | 44 | C2 |        |
| A0> | 33 | 60 | CD | 36 | 44 | C2 | 33 | 60 | 21 | 0D | 62 | 11 | 0D | 61 | 01 | 00 |        |
| B0> | 01 | ED | B0 | D1 | C1 | 2A | 13 | 52 | B7 | ED | 42 | C8 | 03 | CD | 42 | 44 |        |

FED/CMD Drive 5 Record 13 X'0000D' Relative Byte >82  
 Command: R Values X'61'=97

## ENTERING FED

To enter FED, simply type FED <ENTER> at the LDOS Ready prompt. Doing so will cause FED to be loaded and executed. The first prompt you will see will ask you to enter a filespec. Answer this prompt by giving the filespec of the file you wish to examine/modify. If you wish to exit FED at this point, press the <BREAK> key, and you will be returned to the LDOS Ready prompt. If an illegal or improper filespec is given, the appropriate error message will appear, and you will be allowed to re-enter the filespec. The filespec prompt may be bypassed by entering FED using the syntax: FED filespec<ENTER>.

After a valid filespec has been given, the FED 256 character mode will appear on the screen, and the first record (record 0) will be contained in the "edit buffer" (The term "edit buffer" will refer to the record of the file currently in the computer's memory. The edit buffer will contain one 256 byte record at any given time). There will be two cursors flashing within the record (one cursor will be in the "ASCII" portion of the screen, the other cursor will be in the "Hex" display portion), and upon initially accessing a file, these cursors will be positioned over relative byte X'00' of record X'0000'. Throughout this documentation, the term "relative byte" will be used, and will indicate the byte number (0-255) relative to the sector in question. Also, hexadecimal notation (X'nn') will be used to represent the current record number and relative byte number.

There will also be an input cursor located on the bottom right portion of the screen, following the message "Command". This will be referred to as the "command buffer", and will be the place on the screen where commands are entered. The current command in use will always be displayed there. When in the 128 character mode, the command buffer will appear on the lower left portion of the screen.

Also shown on the screen will be additional information which may be of importance to the user (such as current record number, filespec, relative byte within the sector, etc.). The sample displays on the previous page will show where on the screen this information will be displayed. For certain commands, inputs of several characters will be required. Depending on the mode you are in (256 or 128 character mode), these inputs will be taken in a different manner.

When in the 256 character mode, these types of inputs will be taken in an input box, and the input box will be positioned vertically along the right hand edge of the display.

When in the 128 character mode, these types of inputs will be taken directly to the right of the command buffer. No input box will appear, but a flashing cursor will be present, indicating that an input is requested.

It is advised that when using FED, the <BREAK> key should always remain enabled, as some FED commands are exited by the use of the <BREAK> key.

The remainder of this manual will be dedicated to the discussion and explanation of all commands available in the FED program.

## FED LIBRARY

<A> Enter ASCII character modify mode  
<B> Position to the Beginning record  
<C> cccccc ASCII Character string search for cccccc  
<D> Dump Disk File to printer (from current position)  
<E> Position to the Ending record  
<F> nnnnnn Find Hex string nnnnnn  
<G> Go to the next occurrence of last search (Hex or ASCII)  
<H> Enter Hex modify mode  
<L> nnnn Locate Hex load address nnnn  
<M> Memory location of a specified byte  
<N><ENTER> New File request (open a different file)  
<O> Output a top-of-form to printer (X'0C')  
<P> Print current record in edit buffer  
<R> nnnn Position to Record nnnn  
<S><ENTER> Save current record (sector) in edit buffer  
<T> Toggle between 256 and 128 display mode  
<X><ENTER> eXit FED and return to LDOS Ready  
<Z> "Zip" through File Load Blocks  
<BREAK> Cancel current FED command  
<ENTER> Display FED instruction set (Menu)  
<;> (+) Advance one record in the file  
<-> Backup one record in the file  
<SHIFT><=> Display binary representation of byte (128 byte mode only)

## CURSOR MOVEMENT

<←> Move cursor left.  
<→> Move cursor right.  
<↑> Move cursor up.  
<↓> Move cursor down.  
<SHIFT><↑> Position cursor to relative byte X'00' of the current record.

## MENU DISPLAY OF FED INSTRUCTION SET

|                              |                          |
|------------------------------|--------------------------|
| <;> Forward ONE Record       | <BREAK> Cancels command  |
| <-> Backward ONE Record      | <N><ENTER> New File      |
| <B> Beginning Record of File | <S><ENTER> Save Record   |
| <E> Ending Record of File    | <X><ENTER> Exit FED      |
| <R> Position to Record       | <H> Hexadecimal Modify   |
| <Z> Go to next Load Block    | <A> ASCII Modify         |
| <M> Calculate Load Address   | <T> Toggle Display modes |
| <C> Find ASCII String        | <F> Find Hex string      |
| <L> Locate Hex Load Address  | <G> Go next occurrence   |
| <D> Dump File to Printer     | <O> Output top-of-form   |
| <P> Send Buffer to Printer   | <=> Display Binary Value |

Press <ENTER> to Return to Display Mode

## FED MANIPULATION COMMANDS

- <;> Advance one record sequentially in the file. For example, if FED was currently displaying record X'000C' and <;> was pressed, the contents of record X'000D' would be displayed (provided that a record X'000D' existed in the file). An "\*" will be displayed directly below the record number when pointing to the last record in the file. Issuing the <;> command will not change the position of the relative byte cursors. A "+" will be shown in the command buffer to show positive motion in the file.
- <-> Back up one record in the file. If FED was currently displaying record X'0087' and <-> was pressed, the contents of record X'0086' would be displayed. Issuing the <-> command does not change the position of the relative byte cursors. The <-> command will be ignored if it is issued when record 0 is being displayed. A "-" will be shown in the command buffer to show negative motion in the file.
- <B> Position to the beginning of the file (record X'0000') and point cursors to relative byte X'00'.
- <E> Position to the Ending record of the file. An "\*" will appear directly below the record number, indicating that the record being displayed is the last record in the file. The relative byte cursors will be positioned on the last byte in the file (not necessarily relative byte X'FF'). Since LDOS uses sector I/O, the whole sector will be displayed, and any byte in the sector may be modified. Realize that any modifications made to bytes beyond the last byte will not cause the EOF marker of the file to be updated to reflect these changes.
- <R>nnnn Position to record X'nnnn', provided record X'nnnn' exists in the file. If the record does not exist, an "\*" will appear in the command buffer. After entering <R>, a box will appear below the record number display box. The input for the record number to retrieve will be taken in this box. Hex digits (0-F) must be entered, as any other characters will be ignored. You may press <BREAK> to cancel this command. The user may enter the record number without using the standard four digit (X'nnnn') format. Simply type in the record number and press <ENTER>. For example, if the desired record number is X'0021', type <R> <2> <1> <ENTER>. To position to record X'0007', type <R> <7> <ENTER>. The position of the relative byte cursors will remain unchanged after the new record is retrieved.

<Z>

Points the cursors to the next "Type" byte (X'01', X'02', X'05', X'07', X'10', X'1F') of a Load Module File. This feature is designed to allow the user to ZIP through machine language files quickly. Place the cursors on a "Type" byte and press <Z>. After this has been done, the cursors will be positioned over the next "Type" byte. Encountering a X'02' will terminate a <Z>ip. Any string searching, address locating, or address calculating will disable an active <Z>ip. For more information on "Type" bytes, refer to FILE FORMATS in the Technical Information section of the LDOS manual.

### FED MODIFICATION COMMANDS

<A>

Enters the ASCII Modify Mode. In this mode, modifications can be made in ASCII. Anything you can type in from the keyboard (with the exceptions of the <BREAK> key and the arrow keys) can be sent to the edit buffer. Modifications can be made by positioning the cursor over the bytes to be changed. After the A command is issued, the command buffer will display an "A". From this point on, any characters entered will be taken as modifications to the bytes in the record. The arrow keys may be used to position the cursor for additional edits. To exit the ASCII modify mode, the <BREAK> key must be pressed.

To modify a byte: 1) Position the cursor to the desired byte to change. 2) Type in the ASCII character to replace the original. After making a modification, the relative byte cursors will move to the next byte of the record. Note - no changes are made to disk, only to the edit buffer. To make changes to disk, see the <S>ave command.

<H>

Enter the Hex Modify Mode. In this mode, the user can modify bytes in the currently displayed record. Modifications can be made by positioning the cursor over the bytes to be changed. After the H command is issued, the command buffer will display an "H". From this point on, any characters entered will be taken as modifications to the bytes in the record. The arrow keys may be used to position the cursor for additional edits. To exit the Hex modify mode, the <BREAK> key must be pressed.

To modify a byte: 1) Position the cursor over the desired relative byte in the record. 2) Enter the hex digits that you wish to overwrite the current information with. As digits are entered, the previous hex digits will be replaced by the digits entered from the keyboard. The first hex digit entered will modify the first hex digit in the byte, and the second hex digit entered will modify the second hex digit in the byte. After an entire byte has been modified, the cursors will move to the next byte in the record. Note - no changes are made to the disk, only to the edit buffer. To make changes to disk, see <S>ave.

<S><ENTER> Save the contents of the current edit buffer to disk. The current record pointed to by FED will be overwritten by the contents of the edit buffer. Any changes made after the initial read of the record will be written to disk.

#### FED SEARCH COMMANDS

NOTE: The search commands described below may cause the information in the edit buffer to be overwritten by information contained in subsequent records of the file. If edits have been made to the information in the edit buffer, they should be saved to the disk prior to issuing a search command. In most cases, you should issue a "B" command prior to performing a search. This will assure that the entire file will be searched, and no occurrences of the search string will be missed.

<C>cccccc Find ASCII string "cccccc". Issuing the <C> command will cause a search to be performed for the string (cccccc). The search will start at the relative byte pointed to by the cursors. The search is identical to the <F>ind Hex string command, except that the search criteria is an ASCII string of 1 to 30 characters (depending on the display mode being used). Also, the number of characters to be searched for may be an even or an odd number. See the <F> command for further information.

<F>nnnnnn Find hex string "nn nn nn". The <F> command will perform a search for the hex string nn nn nn, starting at the relative byte pointed to by the cursors. (If in the 256 byte display mode, the length of the hex string may be from 2 to 6 characters long, and must be represented as an even number of characters. If in the 128 byte display mode, the length of the hex string may be from 2 to 30 characters long, and must be represented as an even number of characters). The search will begin from the byte over which the cursor is positioned, and will scan all records past the current record until the first occurrence of the string is encountered. If a match is found, the record containing the match will be displayed, and the cursors will be positioned over the first character of the record which matches the search string. To terminate any search, you may press the <BREAK> key. This will cause the record which was contained in the edit buffer prior to the search to be read back in from the disk. If a match is not found, an "\*" will appear in the command buffer, and the cursor will be positioned over relative byte X'FF' of the last record. Only hex bytes can be entered, not hex digits. An "\*" will appear in the command buffer if an odd number of hex digits are entered. If there are multiple occurrences of the specified string, you can "go" to each occurrence by means of the <G>o command.

<G> Go to the next occurrence of current search criteria (string or "L" address). The <G>o command performs a continuation of the last search. If the last search was for a string, it will go to the next occurrence of that string. If the last search was for an address, it will <G>o to the next occurrence of that address. Note - the <G>o works in conjunction with the last search! If the data searched for is not found, one of two things will happen. If the <G>o command is issued after an <L> command and the address is not located, the current record will be read in from disk, and the position of the relative cursors will be unaffected. If the <G>o command is issued after any other search command and the search criteria is not located, the last record will be displayed with the cursor pointing at relative byte X'FF'.

<L>nnnn Locate Hex load address X'nnnn'. The <L> command allows the user to find load address X'nnnn' in a load module file. Unlike the string searches, the <L>ocate command starts its search at record X'0000', rather than at the current cursor position. If the address is located, the record containing the byte at that load address will be displayed, and the cursors will be positioned over this byte. If the address is not located, an error message will be displayed, and you will be prompted to press <ENTER> to continue. After <ENTER> is pressed, the record which was in the edit buffer prior to issuing the <L> command will be retrieved, and the position of the cursors will be unaffected. If a <L>ocate is performed on a non-load module file, the appropriate error message will be displayed. The <G>o command may also be used in conjunction with the <L> command to locate multiple occurrences of the same load address.

#### FED OUTPUT COMMANDS

<D> List the file to the printer, in the same format as the <P>rint command. The <D> command will print all records in the file, starting from the current record number. All records to be printed will be read in from the disk. To halt the printing prior to its completion, depress the <BREAK> key. After the printing has been completed (or terminated), the record which was in the edit buffer prior to printing will be retrieved from disk and stored in the edit buffer, and the cursor position will remain unaffected. Realize that if changes have been made to the record in the edit buffer, these changes should be saved to the disk prior to issuing the <D> command. Several precautions have been taken to prevent computer lock-up during the printing of records. If the printer should become disabled for some reason during printing, FED will continue the printing process after the printer has been enabled. Please note that the LDOS spooler will work in conjunction with the printing operations of FED. Also note that all records will be printed in 20 lines, with a spacing of 2 lines between records. This will allow 3 records to be printed on 66 line/page paper.

- <O>           Output a top-of-form character (X'0C') to the printer.
- <P>           Send edit buffer contents to a printer in ASCII and Hex. The <P> command will print the contents of the edit buffer. After the <P> command has been issued, the record display on the screen will be sent to the printer. To terminate printing at any time, depress the <BREAK> key. The following is a sample of the output produced by the <P> command:

```

                SPACE/CMD  DRIVE 1  RECORD 22  X'0016'

0123456789ABCDEF  BYTE  00 01 02 03 04 05 06 07   08 09 0A 0B 0C 0D 0E 0F
=====
<.2<...<2....D.  <00>  3C 09 32 3C 7F 3A 04 7F   3C 32 04 7F C3 F6 44 A5
GAME OVER PLAYER  <10>  47 41 4D 45 20 4F 56 45   52 20 50 4C 41 59 45 52
< >NEW HIGH SCO  <20>  20 3C 20 3E 4E 45 57 20   48 49 47 48 20 53 43 4F
REEN....TER NAME  <30>  52 45 45 4E 01 00 B4 97   54 45 52 20 4E 41 4D 45
                !.  <40>  20 20 20 20 20 20 20 20   20 20 20 20 20 20 21 C4
W:`...Ww:a...Ww:  <50>  57 3A 60 7F CD F1 57 77   3A 61 7F CD F7 57 77 3A
a...Ww:b...Ww:b.  <60>  61 7F CD F1 57 77 3A 62   7F CD F7 57 77 3A 62 7F
..Ww....0#.././.. <70>  CD F1 57 77 C9 E6 0F C6   30 23 C9 CB 2F CB 2F CB
/./...0#.  PLAYE  <80>  2F CB 2F 18 F0 C6 30 23   C9 20 20 50 4C 41 59 45
R < > .....  <90>  52 20 3C 20 3E 20 20 FF   FF FF FF FF FF FF FF FF
.....  <A0>  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
.....  <B0>  FF FF FF FF 80 88 B7 B7   B7 B7 9D 80 AE BB BB BB
..... INTRU  <C0>  BB 84 80 80 80 80 80 80   80 80 20 49 4E 54 52 55
DERS.....  <D0>  44 45 52 53 AE 9D AE 9D   88 9B A7 84 88 9E AD 84
....0xH.x.0|H0.H  <E0>  A0 99 A6 90 30 78 48 B4   78 84 30 7C 48 30 F8 48
POINTS20 POINTS1  <F0>  50 4F 49 4E 54 53 32 30   20 50 4F 49 4E 54 53 31

```

#### FED MISCELLANEOUS COMMANDS

- <ENTER>       Display FED instruction menu.
- <X><ENTER>   Exit FED and return to LDOS Ready.
- <N><ENTER>   Open a New file for editing. A prompt for the filespec will be displayed. If you input an invalid or improper filespec, an error message will appear, and you will be allowed to re-enter the filespec. Note that FED will never close files, as files need not be closed with this type of editor.
- <BREAK>       Clear command buffer. Pressing <BREAK> will cancel any partial command, and will cause the termination of any command being executed. It is also the only way to exit the ASCII and Hex modify modes. Anytime there is any doubt as to the operation being performed by FED, you may press <BREAK>, and the command buffer will be cleared.

<SHIFT><=> Display binary representation of byte pointed to by the cursors. This command may only be used when in the 128 character mode, and will be ignored if issued in the 256 character mode. After depressing <SHIFT><=>, 8 binary digits will be displayed next to the command buffer. For example, if the cursors were positioned over relative byte X'27', and this byte of the edit buffer contained a X'F3', the binary digits 1111,0011 would be displayed.

<M> Calculates the address in memory where the byte pointed to by the cursors will load. This command works with load module format files only. If the byte is contained in a load block, the load address will be displayed below the record number. If the byte is not in a load block (e.g. a comment line, file header, etc.) the error message "Byte not in load block" will be displayed.

<T> Toggle between the regular 256 byte mode and the extended 128 byte mode. By pressing <T>, the user shifts to "the other" mode. The 128 character mode has all of the same commands as the 256 character mode. The display is a window of the 256 byte record, and 128 bytes will be displayed. By moving the cursors (usually with the <UP> and <DOWN> arrows) you will notice a scrolling effect. The ASCII display will be at the top of the screen instead of the 16 leftmost columns. The current record number is displayed in decimal as well as hexadecimal. All inputs will be taken horizontally instead of vertically. ASCII and hex search inputs will allow 30 characters instead of 6.

#### USING DRIVERS, FILTERS, OR PROGRAMS WITH FED

FED works harmoniously with other programs as long as FED is not tampered with. User programs should not use any memory below X'7700'. When returning to FED from some other function, the display may appear to be garbaged. Simply press <BREAK> and the FED display will be re-established. As far as drivers and filters are concerned, FED uses keyboard, video and printer DCB'S, so any vectors changed by another driver will be picked up by FED. NOTE: For FED to function properly, the user must maintain standard ASCII values and restore any registers, DCB's, devices, etc. to their original values. You may also use FED in conjunction with a machine language program. It is possible (although not recommended) to use FED from within LBASIC via the CMD"FED" command. In order to utilize FED in this manner, the user should make sure that at least 10,000 bytes are free in LBASIC. If the number of free bytes is less than 10,000, the system will most likely crash.

X'nnnn'=  
Drr,bb=  
Lnn

The Lnn line is used to identify a particular library command module, and should not be used by the user.

The X'nnnn'= and Drr,bb= are used to identify the patch line as either a patch by memory load location or a direct disk modify patch, respectively. Information following the = sign will be the actual patch code. It must be entered in one of two ways:

It may be entered as a series of hexadecimal bytes separated by a single space.

It may be entered as a string of ASCII characters enclosed in quotes.

No matter which method is used, there is never a space left between the = sign and the start of the patch code.

### LDOS PATCH MODES

X'nnnn'=nn nn nn nn nn nn .....  
X'nnnn'="String"

This type of patch will patch a file by memory load location. The patch code will be written into a load module added to the end of the file being patched. This ending module will then load with the program and overlay or extend the code at X'nnnn', where nnnn is the memory load address for the patch code. The patch code can be entered either as hexadecimal bytes, or may be represented as an ASCII string. It must be noted that this patch mode will extend the disk file, even if all of the patching is to the "inside" of the program. Because this type of patch will merely be added to the end of the file to be patched, it may later be removed with the YANK parameter.

Drr,bb=nn nn nn nn nn nn .....  
Drr,bb="String"

This is the direct disk modify patch mode. The rr represents the record number in the file to be patched, and the bb is the byte in that record where the patch is to begin. Again, the actual patch code can be either hexadecimal bytes or an ASCII string. This type of patch line does not extend the file and is applied directly to the record of the file. Because no identification of the existence of this patch will be placed in the file, this type of patch cannot be removed by the YANK parameter.

The LIST library command with the (HEX) parameter can be used to display a file, showing the record number and the offset byte. This is an easy way to find the location in the file you wish to patch. Be aware that the first record in a file will be record 0, not record 1.

Lnn

This format is the indicator that the patch code that follows will be to either the SYS6/SYS or SYS7/SYS library command module. The "Lnn" represents the binary coded location of the desired overlay in the SYS module. The patch code that follows will be in either the X'nnnn' or Drr,bb format.

NOTE: This type of PATCH should not normally be created by the user. Any necessary patches to library commands will be issued by Customer Service.

## (YANK)

The patch (YANK) parameter will allow you to remove patches applied with the X'nnnn' format. The following rules will be in effect:

- 1) The filespec of the patch to YANK must be identical to the filespec used when the patch was applied.
- 2) If YANK is used without a filespec, no patch will be removed.
- 3) DO NOT PATCH A FILE MORE THAN ONCE USING THE SAME FILESPEC FOR THE PATCH FILE! It will be impossible to YANK the second patch from the file.

Here are some examples that will show the different patch formats.

```
PATCH BACKUP/CMD:Ø USING SPECIAL/FIX:1
PATCH BACKUP SPECIAL
```

These commands would produce identical results. The default file extensions are /CMD for the file to be patched, and /FIX for the file containing the PATCH information. The patch information in SPECIAL/FIX might look like this:

```
.SPECIAL PATCH FOR MY BACKUP SYSTEM ONLY!
X'6178'=23 3E 87
X'61AØ'=FF ØØ ØØ
```

This is an example of a patch using the X'nnnn' load location format. Note the comment line in the patch code file. This line will have no effect on the patch.

```
PATCH SYS2/SYS.PASSWORD USING TEST/FIX
PATCH SYS2/SYS.PASSWORD TEST
```

Note the abbreviated syntax of the second example. The USING and default /FIX extension are not necessary. The information in the patch file TEST/FIX might look like this:

```
.This will modify the SYS2 Module
DØB,49=EF CD 44 65
DØB,52=C3 ØØ ØØ
.EOP
```

This is an example of the direct patch mode. It will patch the specified record and byte in the file SYS2/SYS. There are 2 comment lines in this patch file. Neither will have any effect on the patch.

```
PATCH SYS6/SYS LIB1
```

This command will patch the SYS6 Library module. The patch file LIB1/FIX might contain the following information:

```
L54
X'52Ø8'=32 2Ø DE AF ØØ C3 66 ØØ
```

This patch is in the memory load location mode. Library patches may also be done with the direct disk modify mode.

## P D U B L / C M D

PDUBL is a disk driver program for use with the Model I, 5" drives, and a double density modification board other than the Radio Shack board. The syntax is:

```
=====
| PDUBL                                     |
|                                         |
| No parameters are required.             |
|                                         |
=====
```

This command loads a special disk driver program which allows you to use a double density hardware modification to read, write, and format double or single density 5" disks with the Model I. Before buying a double density board, please check with the manufacturer or LDOS Support to assure compatibility with the LDOS PDUBL driver.

If you have a doubler installed, after you give this command, you can use either single or double density disks in any of your 5" disk drives. LDOS will automatically recognize whether you have a single or double density diskette in a drive, and react accordingly. Once you have installed the PDUBL driver, you will see the prompt "Single or Double density <S,D> ?" appear after you enter the disk name and master password during the disk FORMAT utility. Answer this prompt by pressing the <D> key to create a double density diskette or <S> to create a single density diskette. Pressing <ENTER> for this prompt will default to double density.

PDUBL also includes support for double-sided 5" drives. Both sides of the diskette are treated as a single volume. The drives and cable must be set up correctly for this feature to work.

The PDUBL driver is loaded into high memory and protects itself by lowering the value stored in the HIGH\$ memory pointer. Logical drives 0-7 are set up to use this driver in place of the normal LDOS single density driver. You can use the SYSTEM (SYSGEN) command to save the driver in your configuration file, to be loaded automatically every time you boot. Be sure that any application programs you are using respect the HIGH\$ pointer.

Please note that you CANNOT boot up on a double density LDOS diskette when using a doubler. You may, however, boot up on a single density diskette and exchange it for a double density diskette as soon as the bootstrap operation has finished.



## Q F B

This utility is designed to allow for a backup with format to be performed. Only floppy drives may be used, and the backup performed must be mirror image. The syntax is:

```
=====
QFB :s :d (parm,parm,parm)

:s   is the Source drive. The colon is optional.
:d   is the Destination drive. The colon is optional.

The following optional parameters may be used:

ALL=  parameter used to specify whether all cylinders
      of the source disk will be read and copied to
      the destination disk, or only allocated
      cylinders will be used. The switch ON or OFF
      may be specified, with the default being OFF.

V1=   parameter used to specify whether or not a
      verify of the destination disk is to be
      performed on the 1st pass. The switch ON or OFF
      may be used, with the default being ON.

V2=   parameter used to specify whether or not a
      verify of the destination disk is to be
      performed on the 2nd pass. The switch ON or OFF
      may be used, with the default being OFF.

QUERY= Query for parameters not specified. The switch
        ON or OFF may be used. The default is OFF

abbr:  ON=Y, OFF=N, QUERY=Q, ALL=A
=====
```

The QFB (Quick Format and Backup) utility will allow for the creation of a mirror image backup of a source disk without having to format the destination disk prior to executing the backup. The normal means by which a mirror image backup is made using LDOS is to first format a diskette using the FORMAT utility, and then use the BACKUP utility to perform the backup. The limitations of the QFB utility are as follows:

- 1.) Two distinct floppy drives must be used.
- 2.) The source diskette must have been formatted using the LDOS 5.1.x FORMAT utility, and cannot contain any non-standard format.
- 3.) QFB will run exclusively on LDOS 5.1.x, versions 5.1.3 or later.

QFB will perform a "single pass" format and backup. If QFB is entered with no drives specified, prompts will appear for them. If drive numbers are specified, the first drive number will represent the source drive, and the destination drive will be the second drive number. If no parameters are specified, the defaults will be used.

Consider the results of entering the following command.

QFB 1 2

Drive 1 will be used as the source drive, while drive 2 will be the destination drive. Prior to QFB performing any action, a prompt will appear to load the diskettes. Once the proper diskettes have been installed, press <ENTER>, and the backup will begin. The following actions will take place.

- 1.) The source diskette will be logged in, to determine the type of format.
- 2.) Cylinder 0 of the destination diskette will be formatted.
- 3.) If cylinder 0 of the source disk contains data, it will be read into memory.
- 4.) If cylinder 0 of the source diskette contains data, the information stored in memory (see Step 3) will be written out to the destination diskette.
- 5.) Cylinder 0 of the destination diskette will be verified.
- 6.) Steps 2-5 will be repeated for all remaining cylinders.
- 7.) The following message will appear after the last cylinder has been verified:

Duplication complete      1 disk created

Replace destination disks and press <ENTER> to repeat  
 ..<R> to restart with new parameters  
 ...or....<BREAK> to exit program.

- 8.) Press <ENTER> in response to this prompt to make another mirror image backup.  
 Press <BREAK> to abort the QFB utility. The following prompt will appear:

Load SYSTEM diskette and hit <ENTER>

Place a system diskette in drive 0 and press <ENTER>, to return to LDOS Ready.

If it is desired to use QFB again with different parameters, press <R> in response to the prompt displayed in step 7. Doing so will cause the drives to be prompted for, and prompts will appear for all parameters.

If QFB is to be restarted, or the command QFB (Q=Y) is entered, the following prompts for the parameters will occur:

Duplicate unallocated tracks? (Y/N)  
 Verify on same pass? (Y/N)  
 Verify on second pass? (Y/N)

The first prompt relates to the ALL parameter. If it is answered with <Y>, all cylinders will be read from the source diskette and written to the destination diskette, regardless of whether or not the cylinder contains information. If this prompt is answered <N>, only cylinders containing information will be read and written.

The next prompt relates to the V1 parameter. If it is answered with <Y>, all cylinders on the destination diskette will be verified immediately after all writes. If answered <N>, no immediate verify will be done.

The final prompt corresponds to the V2 parameter. If it is answered with <Y>, all cylinders on the destination diskette will be verified upon completion of all writing to the diskette. If answered <N>, there will be no second pass verification.

If an error occurs, an appropriate error message will be displayed, and a prompt will appear requesting the course of action that is desired. During any QFB operation, the <BREAK> key will be active, and can be used to abort the process.

### I M P O R T A N T

QFB assumes that a mirror image backup is desired, and performs no check on the destination diskette with respect to the existence of data. Any existing information on a destination diskette will ALWAYS be destroyed. Also, QFB will NOT clear the Mod Flags of files on the source diskette.